

myFMbutler Clip Manager 3

User Manual for Mac OS X

CM3.1M_Jan_09

Table of Contents

1° What does it do?	2
2° What do I need?	2
3° How do I make this work?	2
a. Installation	2
b. Get FileMaker clipboard information into Clip Manager	2
c. Send a Clip Manager clip to the FileMaker clipboard	4
d. Using the Clip Browser	5
e. Editing a Clip Manager clip	7
f. Using the Clip History	8
g. Clip Manager Drag & Drop	9
h. About the Clip Manager interface window	10
i. Clip Manager Preferences	12
j. Clip Manager Advanced Search - Regular Expressions	14
k. Clip Manager & AppleScript	21
4° About the FileMaker clipboard	26

1° What does it do?

myFMbutler Clip Manager is an application that allows you to store FileMaker fields, tables, scripts, script steps, custom functions and layouts in an easy-to-use library. You can copy and store objects that have been copied to FileMaker's own clipboard, and then re-use them later. Clip Manager also offers an integrated "Find & Replace", that allows you to easily make batch changes to the clippings, and then send the updated field, script or layout back to FileMaker's clipboard.

2° What do I need?

FileMaker Pro 8 or 9 Advanced or higher on Mac OS X 10.4.11 or higher.

3° How do I make this work?

a. Installation

Run the Clip Manager Installer:

The program will be installed in:

`/Applications/myFMbutler Clip Manager 3/`

Sample Clips will be installed in:

`~/Documents/Clip Manager Clips/`

b. Get FileMaker clipboard information into Clip Manager

There are two ways to get FileMaker clipboard information in Clip Manager: either automatically using the [Clip History](#), or manually using Clip Manager's 'Get' as described below:

Open Clip Manager. The Clip Contents window will be empty, and the 'Get' and 'Set' buttons will be dimmed. The [Clip Browser](#) will show a Sample Clips folder that you can navigate through.

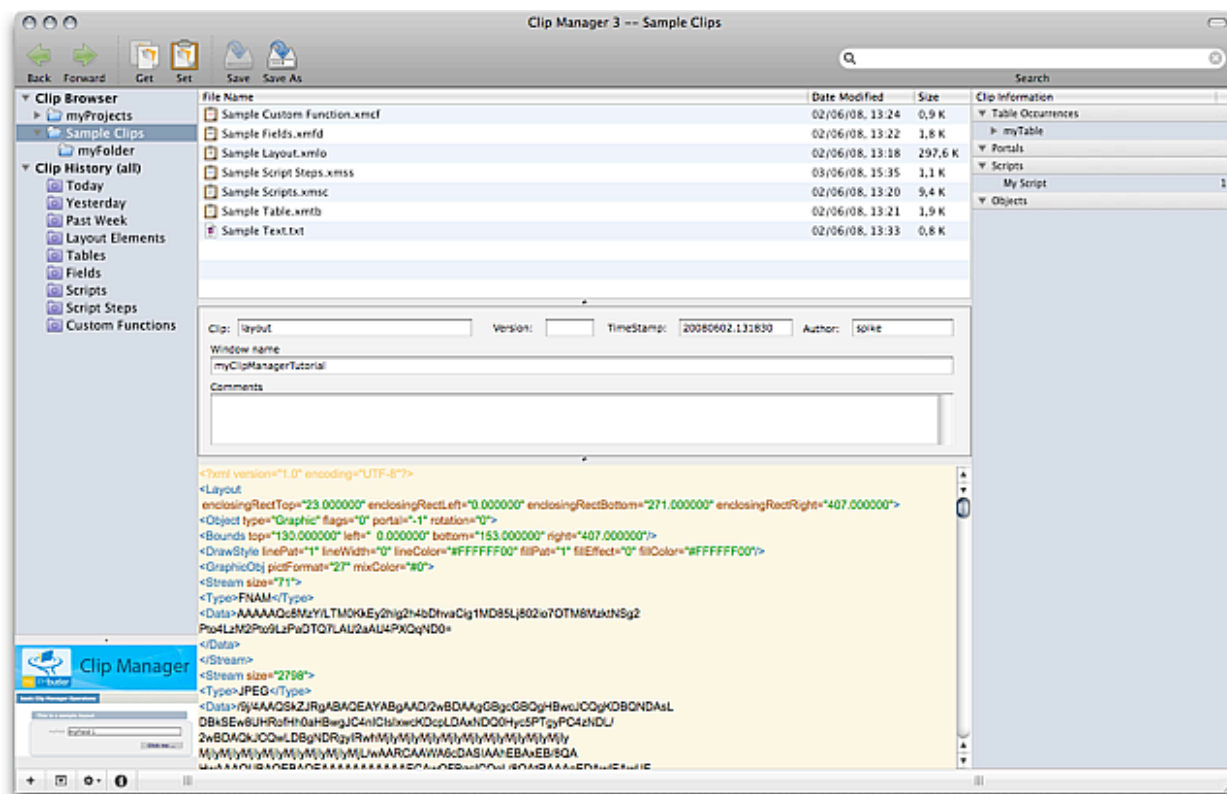
If FileMaker is not yet running, open it first, along with any file of your choice.

Now copy a FileMaker layout object from your sample file, then return to Clip Manager.

You will notice the 'Get' button is now active. Clicking 'Get' or typing cmd-shift-V will create a clip with the FileMaker clipboard information. If the FileMaker clipboard object is a layout, a thumbnail of that layout will also be shown.

You can then add comments for your clipping in the 'Comments' field, and there are a number of extra fields that allow you to store even more metadata in your clip, such as a timestamp, a version number and the author of the clip.

Once you are ready, you can save it to a location of your choice - we suggest in ~/Documents/Clip Manager Clips/, which is the default path to the Clips folder. You can change the location of the Clip Manager clips folder from the 'Clips' tab in Preferences.



This process will be identical for tables, fields, scripts and script steps.

The procedure for Clip Manager to 'Get' a **Custom Function** is a little different, however: you need to make sure that in FileMaker you first open the custom function editor dialog of the function to be copied before going back to Clip Manager.

Important note when "getting" Custom functions:

Because Clip Manager has to drive the user interface to get custom functions, keep the following in mind when using this feature:

- using the keyboard or mouse while Clip Manager is getting the custom function will interfere and produce unpredictable results
- avoid using spaces in custom function names and parameters, Clip Manager sometimes has to perform a double-click to get to these values, and this does not work with strings that have spaces in them.

c. Send a Clip Manager clip to the FileMaker clipboard

Using the browser window, you can navigate through your clip library.

Once you have found the clip file you want to send to FileMaker, double-click to select it, and the 'Set' button will become active. Clicking the 'Set' button or typing cmd-shift-C will send the information in the current clipping to FileMaker's clipboard.

Now return to FileMaker.

Depending on the object type, you can now paste the information:

- for **fields**, go to 'Define Database', and you can 'Paste' the field(s)
- for **tables**, go to 'Define Database', and you can 'Paste' the table(s)
- for **layout objects**, enter 'Layout Mode' and select 'Paste' to paste the objects
- for **scripts**, open 'Script Editor' and you will notice the 'Paste' menu is active
- for **custom functions**, open the 'Define Custom Functions' dialog in FileMaker, and make sure you have no existing custom function selected. Now switch to Clip Manager, select the custom function clip that you would like to copy to FileMaker, and click 'Set'.

If the 'Auto Paste in FileMaker Pro' preference is set, you can also automatically paste layout objects or scripts, if the appropriate window is active and frontmost.

Important note: as FileMaker allows you to copy multiple layout objects, fields, tables or scripts, a single Clip Manager clip can also contain multiple fields etc.

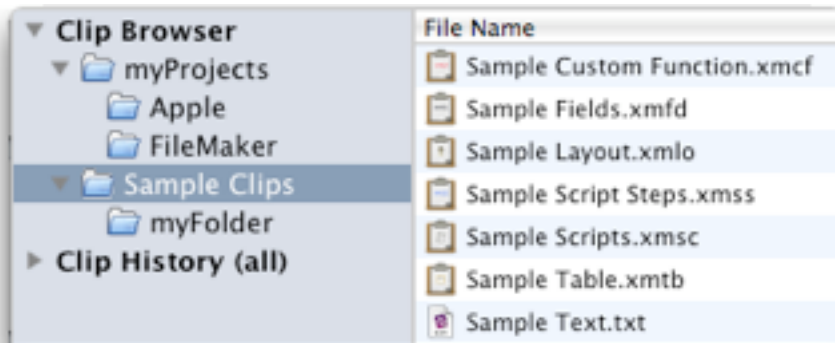
Important note when "setting" Custom functions:

Because Clip Manager has to drive the user interface to set custom functions, keep the following in mind when using this feature:

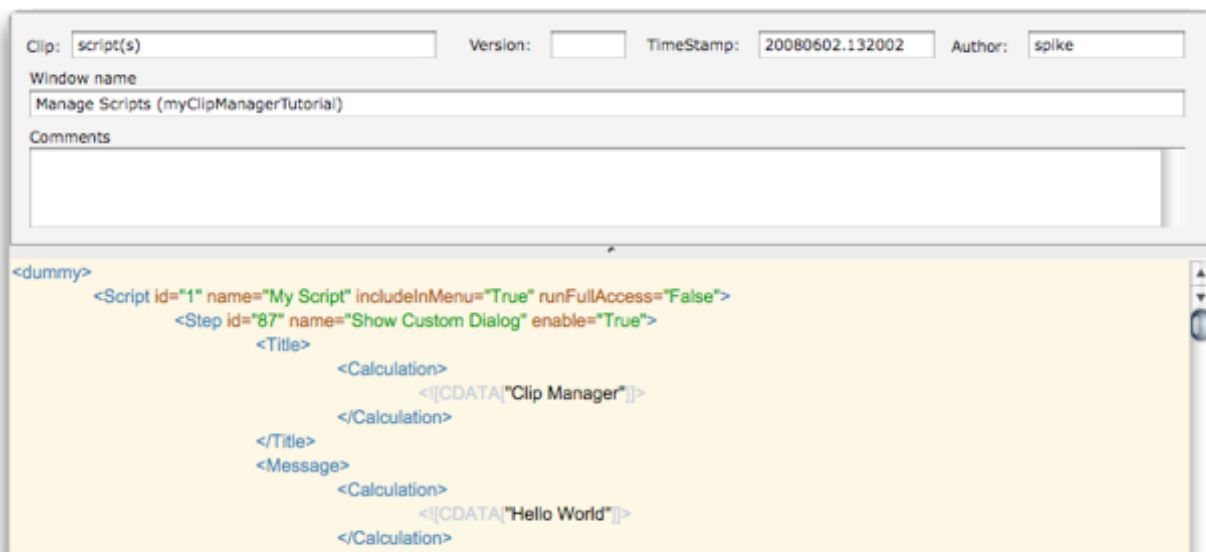
- using the keyboard or mouse while Clip Manager is getting the custom function will interfere and produce unpredictable results
- avoid using spaces in custom function names and parameters, Clip Manager sometimes has to perform a double-click to get to these values, and this does not work with strings that have spaces in them.

d. Using the Clip Browser

The Clip Browser will show all clippings that have been stored in the Clip Manager Clips folder. The default location of the Clips folder is at *~/Documents/Clip Manager Clips/* but you can change this in the Preferences' Clips tab.



When you double-click (or cmd-down arrow) a clip in the browser window, the clip contents and other information from that file will be shown in Clip Manager's contents and metadata fields. You can then use 'Set' to send this clip to the FileMaker clipboard.



If '**Analyze Clips**' is active (which is the default setting), the '**Clip Info**' pane will show information about the tables, fields, portals, scripts, variables and objects that are referenced from the clip.

'Analyze clips' is toggled by clicking the info button on the bottom left of the Clip Manager window.



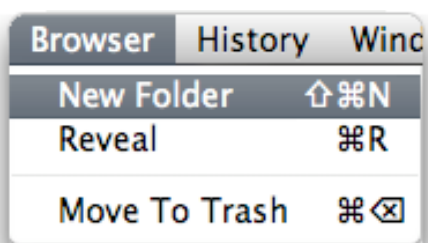
Clip Information	
▼ Table Occurrences	
▼ myTable	
▼ Fields	
myField 1	1
▼ Portals	
▼ Scripts	
My Script	1
▼ Objects	

When double-clicking an item in the Clip Information window, it will be selected in the Clip Contents window.

Important note : due to FileMaker limitations, double-clicking a portal object to select it is not currently supported.

The '**Browser**' menu shows a number of options:

- create a new folder from the Clip Browser
- move a file or folder from the Clip Browser to the Trash
- reveal the current file or folder selection in the Finder



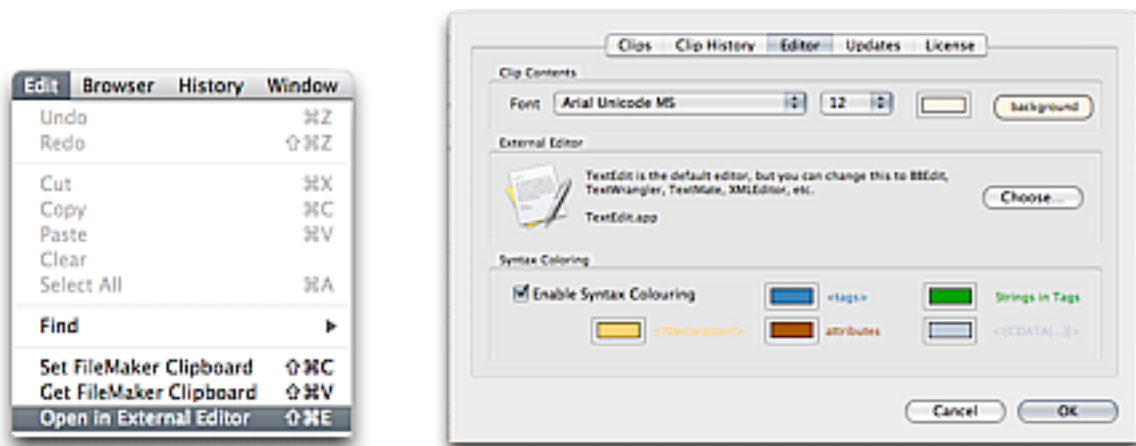
You can store clips by object type, by project etc. You could also create separate clips folders e.g. for easy updating of an existing application...

e. Editing a Clip Manager clip

After performing a 'Get' operation, or when you double-click a clip in the browser window (or use the cmd-down arrow shortcut), the clip contents and other information will be shown in Clip Manager's contents and metadata fields.

Clip Manager provides optional **syntax coloring** to make the XML more readable, and to allow you to make changes more easily. This setting can be configured from the 'Editor' tab in Preferences.

Using '**Find & Replace**', you can easily make changes to the contents of a clip from within Clip Manager. You can access the 'Find' menu item from the Edit menu or use the cmd-F shortcut (make sure you placed your cursor in the Clip Contents field first).



If you need to make more changes than just a basic 'Find & Replace' operation, select 'Open in Text Editor' from the 'Edit' menu (or cmd-shift-E) to open the current clip in a text editor. TextEdit is the default editor, but you can change this in the 'Editor' tab in Preferences. Most popular editors, such as BBEdit, TextWrangler, TextMate, etc. are supported.

The editor makes it easy to perform batch change operations on a clipping file. You can make the changes, then copy-paste the information back to Clip Manager, and then send the clipping back to the FileMaker clipboard by clicking 'Set'.

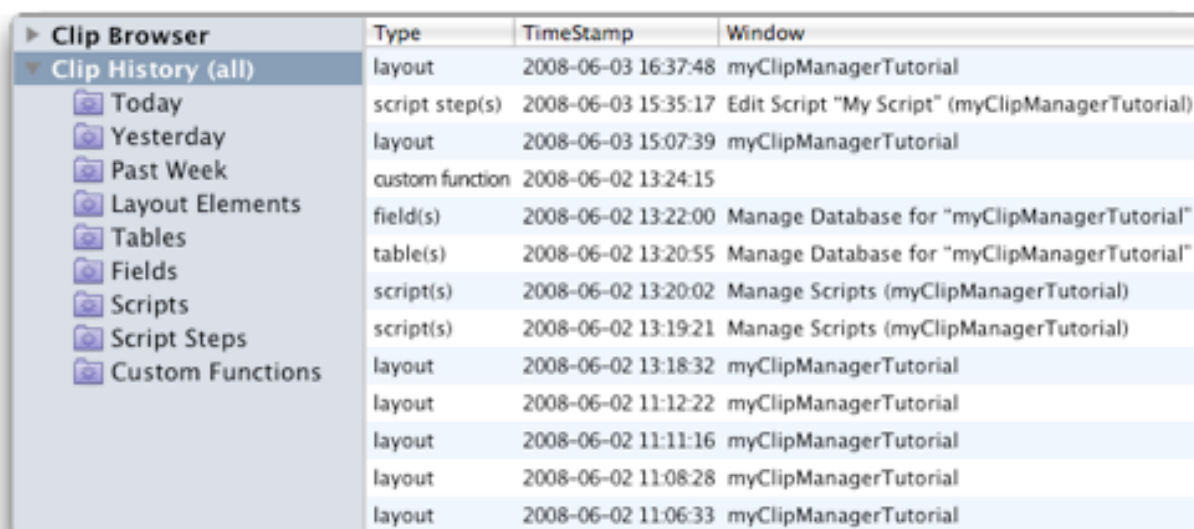
f. Using the Clip History

Clip Manager's Clip History allows you to automatically store all information you copy in FileMaker, i.e. all scripts, script steps, fields, tables and layout elements. Whenever you copy something in FileMaker, Clip Manager will analyze it in the background and add it to a SQLite database.

The Clip History contains a number of **Smart Folders** that should make it easier to quickly find the item you are looking for.

You can use the Clip History for all kinds of purposes, the most obvious being to make sure you have backups when making changes to your solution.

Instead of the usual duplicating of a script when making changes, you can now just hit 'Copy', and work on the original. If it turns out you have to go back to the previous version, select the script(s) from the Clip History list in Clip Manager, and you can set the FileMaker clipboard to paste it back.

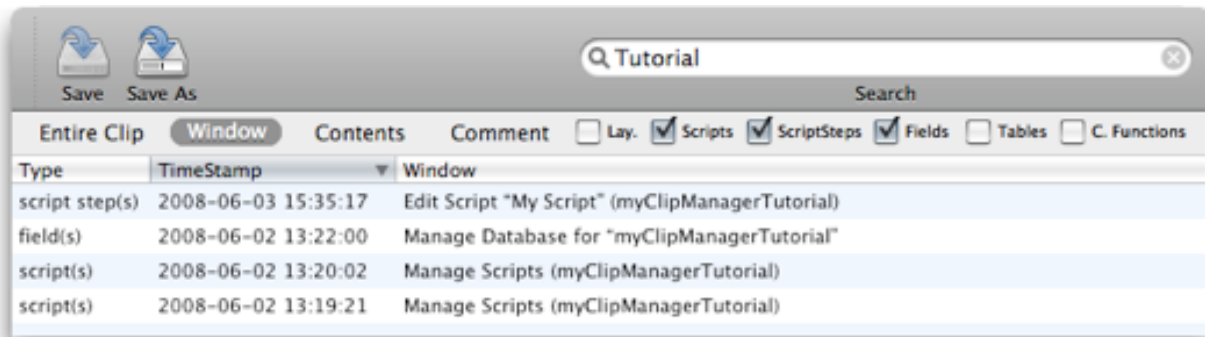


Clip Browser	Type	TimeStamp	Window
Clip History (all)	layout	2008-06-03 16:37:48	myClipManagerTutorial
Today	script step(s)	2008-06-03 15:35:17	Edit Script "My Script" (myClipManagerTutorial)
Yesterday	layout	2008-06-03 15:07:39	myClipManagerTutorial
Past Week	custom function	2008-06-02 13:24:15	
Layout Elements	field(s)	2008-06-02 13:22:00	Manage Database for "myClipManagerTutorial"
Tables	table(s)	2008-06-02 13:20:55	Manage Database for "myClipManagerTutorial"
Fields	script(s)	2008-06-02 13:20:02	Manage Scripts (myClipManagerTutorial)
Scripts	script(s)	2008-06-02 13:19:21	Manage Scripts (myClipManagerTutorial)
Script Steps	layout	2008-06-02 13:18:32	myClipManagerTutorial
Custom Functions	layout	2008-06-02 11:12:22	myClipManagerTutorial
	layout	2008-06-02 11:11:16	myClipManagerTutorial
	layout	2008-06-02 11:08:28	myClipManagerTutorial
	layout	2008-06-02 11:06:33	myClipManagerTutorial

When selecting a clip from the Clip History database, you can also 'Save as' if you want to save the current selection as a standalone Clip Manager clip, that you can then access from the Clip Browser, or e.g. share with someone else.

Instead of constantly backing up a complete solution while working, you can choose to only copy certain scripts, layouts, etc. that you are working on. Extra comment and version fields are provided in the Clip History window. These fields can be edited by double-clicking.

The Clip History Filter allows you to only select a certain FileMaker data type. Using the Search String, a search will be performed according to the criteria you select.



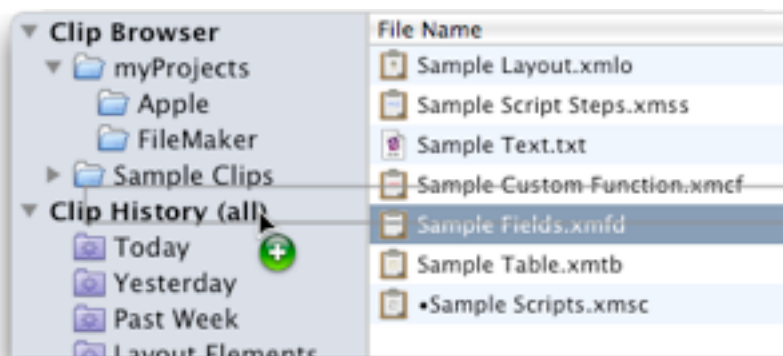
The 'Clip History' tab in Preferences provides lots of configuration options for your Clip History.

g. Clip Manager Drag & Drop

Clip Manager 3 offers extensive Drag & Drop functionality. Here's an overview of how you can use drag & drop to make clip management a lot easier:

- from the clip file list to a browser folder:
 - the clip is moved from the source list to the other folder
 - with option key: the file is copied to the folder
- from the clip file list to the clip history:
 - the file will be copied to the history database.

Note: clip list items cannot be dragged to the clip history smart selectors



- from the clip history to a browser folder:
 - the file will be copied to the folder: a "Save As" dialog will be displayed, and the file will be saved in the folder you select (the default name for a clip history items is "clip history item").

- from the clip history to the Desktop:
 - (for layout items only) the clip is copied to the Desktop as a “Picture clipping”. It is a fully functional FileMaker clipboard item though, and can still be dragged onto FileMaker.
- from the clip file list to the Desktop:
 - the file will be moved to the Desktop.
- from the clip list to FileMaker:
 - (for layout items only) the clip will be copied to the layout.
Note: if the layout area accepting the drop is too small, FileMaker will beep, it will not propose to make the layout bigger.
- from clip preview to FileMaker:
 - (for layout items only) the clip will be copied to the layout.
Note: if the layout area accepting the drop is too small, FileMaker will beep, it will not propose to make the layout bigger.
- from clip preview to the Desktop:
 - (for layout items only) the clip is copied to the Desktop as a “Picture clipping”. It is a fully functional FileMaker clipboard item though, and can still be dragged onto FileMaker.

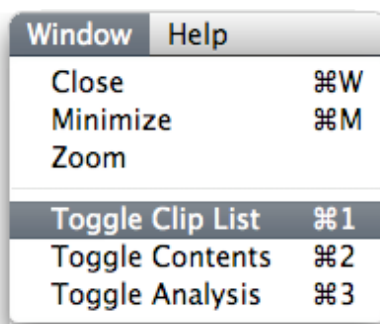
h. About the Clip Manager interface window

The Clip Manager 3 interface offers a lot of customization options, so you can adapt what is shown on screen depending on the type of work you are doing.

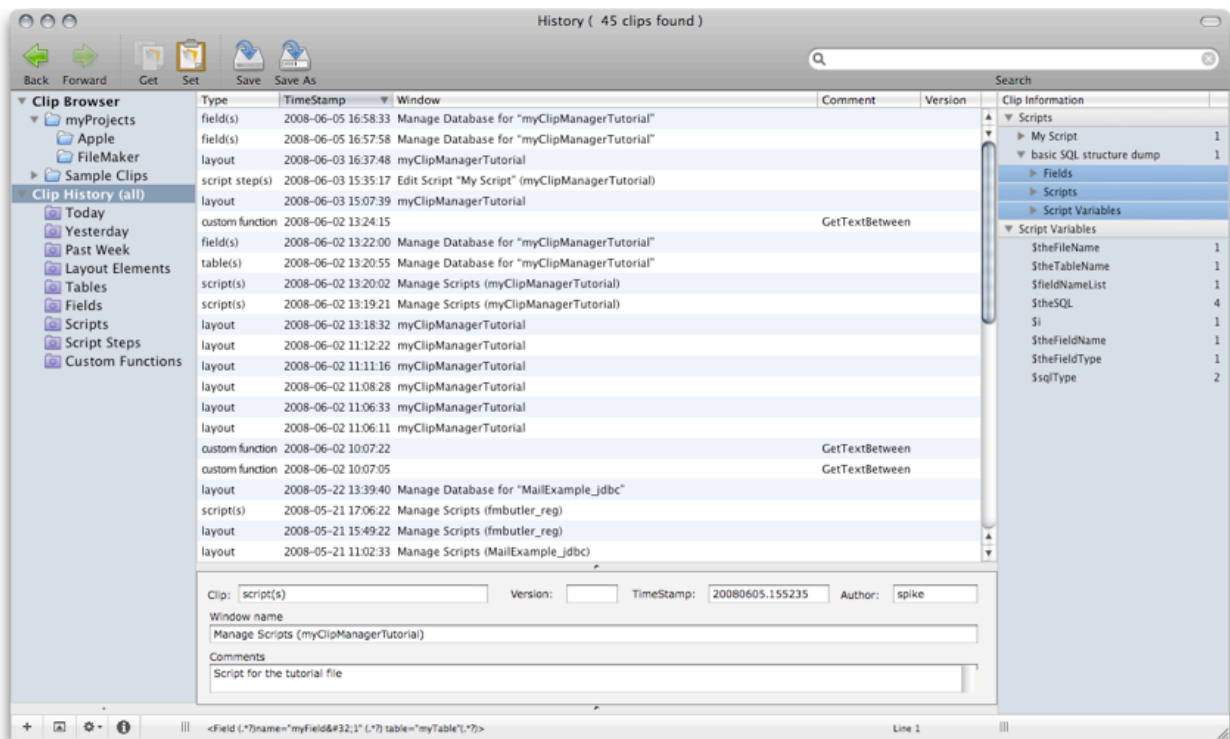
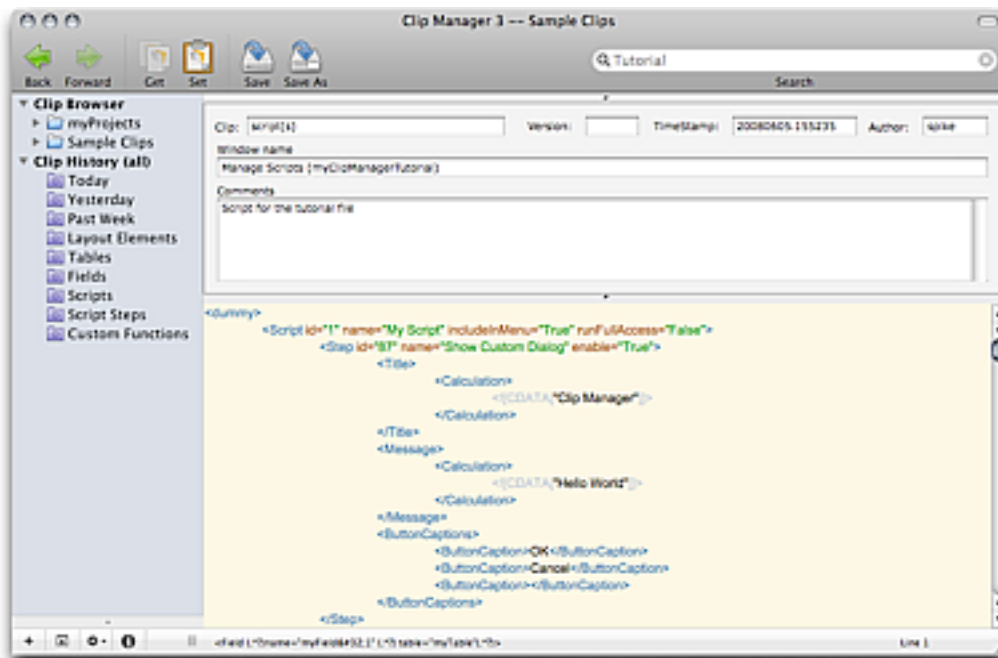
Split screen views allow you to show/hide the following components in the interface window:

- Contents field
- Metadata fields
- Layout preview

You can toggle these options from the Window menu, or you can change the views manually using the split handles, or you can double-click them to completely hide a part.



Below are a couple of examples that show how you can customize the Clip Manager 3 interface window.



i. Clip Manager Preferences

If you select 'Preferences' from the Clip Manager menu, you'll notice 5 tabs:

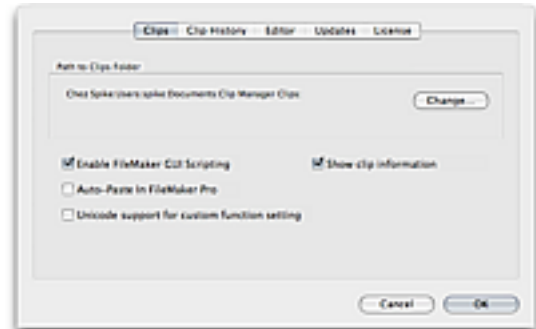
- Clips

Allows you to select the location for your Saved Clips. This can either be a local or a network volume. If you store clips on a network volume, that volume will automatically be mounted when Clip Manager starts up.

'Enable GUI Scripting' is required when you want to 'get' and 'set' Custom Functions.

The 'Auto Paste in FileMaker Pro' option will allow the 'Set' option to not only 'Set' the FileMaker clipboard, but also to automatically 'Paste' in FileMaker if the appropriate window is active, i.e. the 'Script Editor' window for scripts, or a layout in layout mode for layout elements.

'Show clip information' shows more information about the tables, fields, portals, scripts, variables and objects that are referenced from the clip. The 'Unicode support for custom functions' setting should be used with languages with double-byte characters like Japanese, etc.



- Clip History

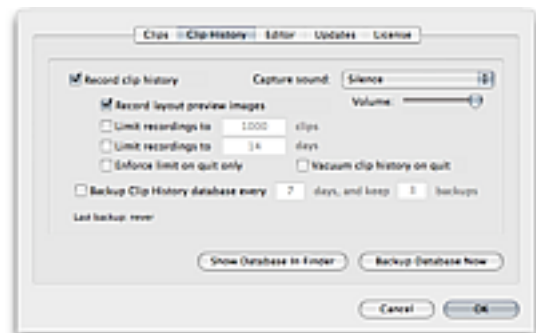
Provides lots of customization options:

You can turn Clip History recording on or off, optionally assign a 'capture sound'.

If you have a slower Mac, you might want to uncheck 'Record layout preview images' to improve Clip Manager background processing.

Limit the number of clips to keep, and for how long to keep them. Optionally perform this check only on quit. The 'Vacuum on quit' option will optimize the SQLite database for better performance.

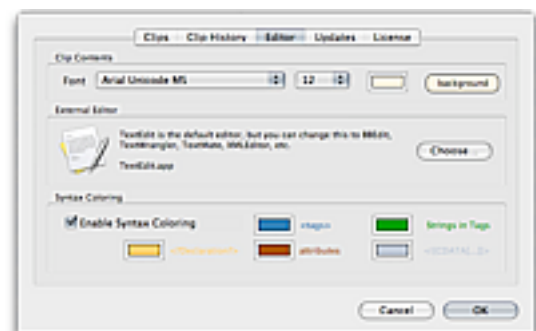
You can manage how to make regular backups of your Clip History.



- Editor

Lets you select the text editor you prefer.

The default editor is TextEdit, but BBedit, TextWrangler, TextMate are popular alternatives. Allows you to toggle syntax coloring in the clip contents window, and select your colors of choice.



- **Updates**

Allows the program to automatically check for updates.

Clip Manager integrates the Sparkle framework for easy updating.

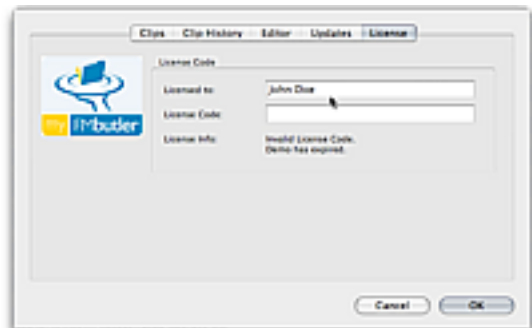
More details at <http://sparkle.andymatuschak.org/>



- **License**

Allows you to register Clip Manager.

The Clip Manager demo version will expire after 30 days.



j. Clip Manager Advanced Search - Regular Expressions

Clip Manager has extensive support for search using regular expressions.

Although plenty of technical information about regex is available on the Internet, here is a concise overview of the search functionality offered in Clip Manager.

Pattern Description

Pattern	Decription
.	Matches any character except newline.
[a-z0-9]	Matches any single character of set.
[^a-z0-9]	Matches any single character not in set.
\d	Matches a digit. Same as [0-9].
\D	Matches a non-digit. Same as [^0-9].
\w	Matches an alphanumeric (word) character -- [a-zA-Z0-9_].
\W	Matches a non-word character [^a-zA-Z0-9_].
\s	Matches a whitespace character (space, tab, newline, etc.).
\S	Matches a non-whitespace character.
\n	Matches a newline (line feed).
\r	Matches a return.
\t	Matches a tab.
\f	Matches a formfeed.
\b	Matches a backspace.
\0	Matches a null character.
\000	Also matches a null character because of the following:
\nnn	Matches an ASCII character of that octal value.
\xnn	Matches an ASCII character of that hexadecimal value.
\cX	Matches an ASCII control character.
\metachar	Matches the meta-character (e.g., \, .,).

Pattern	Decription
(abc)	Used to create subexpressions. Remembers the match for later back-references. Referenced by replacement patterns that use \1, \2, etc.
\1, \2,...	Matches whatever first (second, and so on) of parens matched.
x?	Matches 0 or 1 x's, where x is any of above.
x*	Matches 0 or more x's.
x+	Matches 1 or more x's.
x{m,n}	Matches at least m x's, but no more than n.
abc	Matches all of a, b, and c in order.
alblc	Matches one of a, b, or c.
\b	Matches a word boundary (outside [] only).
\B	Matches a non-word boundary.
^	Anchors match to the beginning of a line or string.
\$	Anchors match to the end of a line or string.

Replacement Patterns

Pattern	Decription
\$`	Replaced with the entire target string before match.
\$&	The entire matched area; this is identical to \0 and \$0.
\$'	Replaced with the entire target string following the matched text.
\$0-\$50	\$0-\$50 evaluate to nothing if the the subexpression corresponding to the number doesn't exist.
\0-\50	
\xnn	Replaced with the character represented by nn in Hex, e.g., TM is TM .
\nnn	Replaced with the character represented by nn in Octal.
\cX	Replaced with the character that is the control version of X, e.g., \cP is DLE, data line escape.

The basic idea of regular expressions is that it enables you to find and replace text that matches the set of conditions you specify. It extends normal Search and Replace with pattern searching.

Wildcards

Some special characters are used to match a class of characters:

Wildcard	Matches
.	Any single character except a line break, including a space.

If you use the "." as the search pattern, you will select the first character in the target string and, if you repeat the search, you will find each successive character, except for Return characters

The following wildcards match by position in a line:

Wild-card	Matches	Example
^	Beginning of a line (unless used in a character class; see below)	^Phone: Finds lines that begin with "Phone":
\$	End of a line (unless used in a character class)	\$. Finds the last character in the current line.

Character Classes

A character class allows you to specify a set or range of characters. You can choose to either match or ignore the character class. The set of characters is enclosed in brackets. If you want to ignore the character class instead of match it, precede it by a caret (^). Here are some examples:

Character class	Matches
[aeiou]	Any one of the characters a, e, i, o, u.
[^aeiou]	Any character except a, e, i, o, u.
[a-e]	Any character in the range a-e, inclusive
[a-zA-Z0-9]	Any alphanumeric character. Note: Case-sensitivity is controlled by the CaseSensitive property of the RegexOptions class.

Character class	Matches
[[]	Finds a [.
[]]	Finds a]. To find a closing bracket, place it immediately after the opening bracket.
[a-e^]	Finds a character in the range a-e or the caret character. To find the caret character, place it anywhere except as the first character after the opening bracket.
[a-c-]	Finds a character in the range a-c or the - sign. To match a -, place it at the beginning or end of the set.

Non-printing Characters

You can use the following notation to find non-printing characters:

Special character	Matches
\r	Line break (return)
\n	Newline (line feed)
\t	Tab
\f	Form feed (page break)
\xNN	Hex code NN.

Other Special Characters

The following patterns are wildcards for the following special characters:

Special character	Matches
\s	Any whitespace character (space, tab, return, linefeed, form feed)
\S	Any non-whitespace character.
\w	Any "word" character (a-z, A-Z, 0-9, and _)
\W	Any "non-word" character (All characters not included by \w).
\d	Any digit [0-9].

Special character	Matches
\D	Any non-digit character.

Repetition Characters

Repetition characters are modifiers that allow you to repeat a specified pattern.

Repetition character	Matches	Examples
*	Zero or more characters.	d* finds no characters, or one or more consecutive "d"s. .* finds an entire line of text, up to but not including the return character.
+	One or more characters.	d+ finds one or more consecutive "d"s. [0-9]+ finds a string of one or more consecutive numbers, such as "90404", "1938", the "32" in "Win32", etc.
?	Zero or one characters.	d? finds no characters or one "d".

Please note that, since * and ? match zero instances of the pattern, they always succeed but may not select any text. You can use them to specify an optional character, as in the examples in the following section.

"Greediness"

Clip Manager supports the "?" as a "greediness" modifier for a subpattern in a regular expression. By default, greediness is controlled by the Greedy property of the RegexOptions class, but can be overridden using the "?". You can place a "?" directly after a * or + to reverse the "greediness" setting. That is, if Greedy is True, using the ? after a * or + causes it to match the minimum number of times possible. For example, consider the following:

Target String	Greedy	Regular Expression	Result
aaaa	TRUE	(a+?) (a+)	\$1=a, \$2=aaa
aaaa	FALSE	(a+?) (a+)	\$1=aaa, \$2=a

Extension Mechanism

We also support the regular expression extension mechanism used in Perl. For instance:

(?#text)	Comment
(?:pattern)	For grouping without creating backreferences
(?=pattern)	A zero-width positive look-ahead assertion. For example, <code>\w+(?=\t)</code> matches a word followed by a tab, without including the tab in <code>\$&</code> .
(?!pattern)	A zero-width negative look-ahead assertion. For example <code>foo(?!bar)/</code> matches any occurrence of "foo" that isn't followed by "bar".
(?<=pattern)	A zero-width positive look-behind assertion. For example, <code>(?<=\t)\w+</code> matches a word that follows a tab, without including the tab in <code>\$&</code> . Works only for fixed-width look-behind.
(?<!pattern)	A zero-width negative look-behind assertion. For example <code>(?<!bar)foo</code> matches any occurrence of "foo" that does not follow "bar". Works only for fixed-width look-behind.

Subexpressions

You can use parentheses within your search patterns to isolate portions of the matched string. You do this when you need to refer to subsections of the matched in your replacement string. For example you would do this if you need to replace only a portion of the matched string or insert other text into the matched string.

Here is an example. If you want to match any date followed by the letters "B.C." you can use the pattern `"\d+\sB\.C\."` (Any number of digits followed by a space character, followed by the letters "B.C.") This will match dates such as 33 B.C., 1742 B.C., etc. However, if you wanted your replacement pattern to leave the year alone but replace the letters with something else, you would use parens. The search pattern `"(\d+)\s(B\.C\.)"` does this.

When you write your replacement pattern, you can refer to the year only with the variable `\1` and the letters with `\2`.

If you write `"(\d+)\s(B.C.|A.D.|BC|AD)"`, then `\2` would contain the matched letters.

Combining Patterns

Much of the power of regular expressions comes from combining these elementary patterns to make up complex searches. Here are some examples:

Pattern	Matches
<code>\\$?[0-9,]+\.\?d*</code>	Matches dollar amounts with an optional dollar sign.

Pattern	Matches
<code>\d+\sB\.C\.</code>	One or more digits followed by a space, followed by "B.C."

The Alternation Operator

The alternation operator (`()`) allows you to match any of a number of patterns using the logical "or" operator. Place it between two existing patterns to match either pattern. You can use more than one alternation operator in a pattern:

Pattern	Matches
<code>\she\s \sshe\s</code>	" he " or " she "
<code>cat dog possum</code>	"cat", "dog", or "possum"
<code>([0-9,]+\sB\.C\.) ([0-9,]+\sA\.D\.) or [0-9,]+\s((B\.C\.) (A\.D\.))</code>	Years of the form "yearNum B.C. or A.D." e.g., "2,175 B.C." or "215 A.D."

Search and Replace

You use special patterns to represent the matched pattern. Using replacement patterns, you can append or prepend the matched pattern with other text.

Pattern	Matches
<code>\$&</code>	Contains the entire matched pattern. If " <code>\d\d\d\d\sB\.C\.</code> " finds "1541 B.C.", then the replacement pattern " <code>the year \$&</code> " results in "the year 1541 B.C.", as the <code>\$&</code> contains the string "1541 B.C".
<code>\1, \2, etc.</code>	Contains the matched subpatterns, defined by use of parentheses in the search string. The search pattern " <code>(\d+)\s(B\.C\. A\.D\. BC AD)</code> " looks for any number of digits followed by a space character, followed by either "B.C.", "BC", "A.D.", or "AD". The <code>\1</code> variable contains the match to the " <code>\d+</code> " portion of the expression and <code>\2</code> contains the match to the " <code>B\.C\. A\.D\. BC AD</code> " portion.

Credits

Clip Manager uses a modified version of the PCRE library package, which is open source software, written by Philip Hazel, and copyrighted by the University of Cambridge, England.

The source to this library is available at:

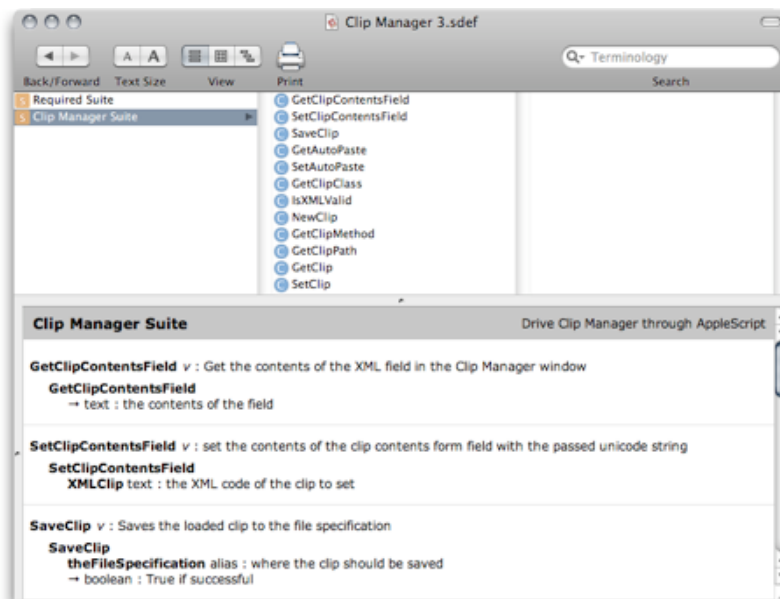
<ftp://ftp.csx.cam.ac.uk/pub/software/programming/pcre/>

The RegEx documentation is based on information from Realbasic:

<http://www.realsoftware.com/products/realbasic/>

k. Clip Manager & AppleScript

Clip Manager has extensive AppleScript support. Open the Clip Manager dictionary in Script Editor for more information. You can do this by dropping the Clip Manager application onto Script Editor.



CopyFromClipContentsField

v : Copies the contents of the XML field in the Clip Manager window to the clipboard

CopyFromClipContentsField

→ text : the contents of the XML field

GetAutoPaste

v : Gets the status of the AutoPaste flag

set theResult to GetAutoPaste

→ boolean : the status

GetClip

v : Gets the currently loaded clip

set theResult to GetClip

→ unicode text : the clip XML contents

GetClipClass

v : Gets the class of the loaded clip

set theResult to GetClipClass

→ unicode text : the class of the clip, being
"XMLO","XMFD","XMTB","XMSC","XMSS" or "XMCF"

GetClipContentsField

v : Gets the contents of the XML field in the Clip Manager window

set theResult to GetClipContentsField

→ unicode text : the contents of the field (should be UTF-8 text)

GetClipMethod

v : Gets the "method" with which the clip was created

set theResult to GetClipMethod

→ text : "new" for a new clip, "file" for a file based clip, "history" for a clip history item
and "session" for a session based clip

GetClipPath

v : Gets the path of a file based clip

set theResult to GetClipPath

→ file : the path of the clip

IsChanged

v : Returns if the loaded clip has been changed or not

set theResult to IsChanged

→ boolean : true is the clip has been modified

IsXMLValid

v : Returns whether the loaded clip is valid or not

set theResult to IsXMLValid

→ boolean : true means the XML is valid

NewClip

v : Clears the currently loaded clip

NewClip

→ creates a new empty clip

OpenClip

v : Loads a clip file into the clip contents field

set theResult to OpenClip *file* (the path to the clip file)

→ boolean : true if the file has been loaded successfully

PasteToClipContentsField

v : Paste the the unicode contents of the clipboard to the XML field in the Clip Manager window

PasteToClipContentsField unicode text

unicode text : the XML code of the clip to set

ReplaceAll

v : Replaces the first string in the given list by the second string in the given list

set theResult to ReplaceAll *unicode text by unicode text*

optional parameters:

CaseSensitive: boolean ⇐

FindEntireWord: boolean ⇐

UsingRegularExpression: boolean ⇐

Greedy: boolean ⇐

DotMatch: boolean ⇐

TreatingTargetAsOneLine: boolean

→ boolean : true if no errors occurred, check with isChanged

SaveClip

v : Gets the "method" with which the clip was created

set theResult to SaveClip *file* (the location of the clip file)

→ boolean : true if successful

SetAutoPaste

v : Sets the status of the AutoPaste flag

SetAutoPaste *boolean*

SetClip

v : Sets the clipboard and the currently loaded clip to the contents

SetClip *unicode text*

unicode text : the XML code of the clip to set

SetClipContentsField

v : Sets the contents of the clip contents form field with the passed unicode string

SetClipContentsField *unicode text*

unicode text : the XML code of the clip to set

SetReplaceOptionCaseSensitive

v : Sets the replace option to case sensitive

SetReplaceOptionCaseSensitive *boolean*

SetReplaceOptionMatchEntireWord

v : Sets the replace option to match for entire word

SetReplaceOptionMatchEntireWord *boolean*

SetReplaceOptionRegex

v : Sets the replace option Regex to true or false

SetReplaceOptionRegex *boolean*

SetReplaceOptionRegexDotMatch

v : Normally the period matches everything except a new line, this option allows it to match new lines.

SetReplaceOptionRegexDotMatch *boolean*

SetReplaceOptionRegexGreedy

v : Greedy means the search finds everything from the beginning of the first delimiter to the end of the last delimiter and everything in-between.

SetReplaceOptionRegexGreedy *boolean*

SetReplaceOptionRegOneLine

v : Ignores internal new lines for purposes of matching against '^' and '\$'.

SetReplaceOptionRegOneLine *boolean*

SetReplaceOptionReset

v : Sets all Replace Options to False.

SetReplaceOptionReset

4° About the FileMaker clipboard

Ever since the arrival of FileMaker Pro 8 Advanced, it became possible to copy scripts, script steps, tables and fields, as well as layout objects. When copied, the information about these items is stored in an XML format. However, FileMaker does not provide a way to make this XML code available as editable text, but instead stores it in clipboard “flavors” that are not directly recognizable by applications outside of FileMaker.

Clip Manager is able to analyze and translate these flavors back and forth, making them available as editable UTF-8 text, and translates this XML text back to the clipboard chunks FileMaker recognizes as its internal clipboard. For layouts, FileMaker also puts a PDF representation of the layout into its clipboard, and this PDF info is saved into the Clip Manager clips as well. It is also used to display the thumbnail of the layout objects.

The following content types are available:

<i>XMTB</i> : FileMaker table(s)	<i>XMLO</i> : FileMaker layout object(s)
<i>XMFD</i> : FileMaker field(s)	<i>XMCF</i> : FileMaker custom function(s)
<i>XMSC</i> : FileMaker script(s)	<i>TEXT</i> : plain text
<i>XMSS</i> : FileMaker script step(s)	

The content type is used as an extension when saving clips to your Clips folder, e.g.

myTable.xmtb

myLayout.xmllo

myCustomFunction.xmcf

etc.

When you have made changes to a clipping, either from within the Clip Manager ‘clip contents’ field, or with content pasted back from your external text editor, Clip Manager will re-evaluate the content to check whether the XML format is still OK. If it cannot find a valid XML content type, Clip Manager will consider the clipping as just plain text.

Logically, if you set the FileMaker clipboard with plain text contents, that information will only become available in FileMaker from the standard OS ‘Paste’ command, and not from within ‘Define database’, ‘Script Editor’ or as a layout object.

Any questions? Contact us via www.myfmbutler.com.

© 2009 myFMbutler